Project Name: FoodieMate

Project description:

FoodieMate is an app that makes ordering food easy and convenient. Whether you're craving pizza, sushi, or anything in between, FoodieMate helps you find the perfect restaurant and place your order hassle-free. With features like user registration, restaurant search, menu viewing, order placement, and secure payment integration, FoodieMate ensures a smooth and enjoyable food ordering experience for everyone. Users can also track the status of their orders and manage their profiles, while restaurants can update their profiles, menus, and track orders seamlessly. Plus, with delivery tracking, users can keep an eye on their food as it makes its way to their doorstep. With FoodieMate, delicious meals are just a tap away!

1. **Backlog Creation and Product Backlog:**
   - ❖ User Registration/Login: Allow users to create accounts or log in easily.
   - ❖ Restaurant Search: Implement a feature where users can search for restaurants based on location or cuisine.
   - ❖ Menu Viewing: Enable users to view menus of different restaurants.
   - ❖ Order Placement: Allow users to select items from the menu and place orders.
   - ❖ Payment Integration: Implement secure payment options for order transactions.
   - ❖ Order Tracking: Provide users with the ability to track the status of their orders.
   - ❖ User Profile Management: Allow users to manage their profiles, update information, and view order history.
   - ❖ Restaurant Management: Create functionality for restaurants to manage their profiles, update menus, and track orders.
   - ❖ Delivery Tracking: Introduce a feature for users to track the delivery of their orders.

2. **Sprint Planning**:
   - ❖ Sprint 1: Goal: User Registration/Login, Restaurant Search, and Menu Viewing.
   - ❖ Sprint 2: Goal: Order Placement and Payment Integration.
   - ❖ Sprint 3: Goal: Order Tracking and User Profile Management.
   - ❖ Sprint 4: Goal: Restaurant Management and Delivery Tracking.

3. **Estimation:**
   - ❖ Assign story points to each task based on its complexity and the effort required.
   - ❖ For example, tasks like User Registration/Login might be simpler and assigned fewer points, while Payment Integration might be more complex and require more points.

4. **Execution:**
   - ❖ Begin each sprint by selecting tasks from the backlog for that sprint.
   - ❖ Developers work on the selected tasks throughout the sprint.
   - ❖ Hold daily stand-up meetings to discuss progress, obstacles, and plans for the day.
   - ❖ Test features as they are developed to ensure they meet requirements and are bug-free.

5. **Review and Retrospective:**
   - ❖ At the end of each sprint:
   - ❖ Review the completed features with stakeholders.
   - ❖ Gather feedback and make any necessary adjustments.
   - ❖ Conduct a retrospective meeting:
   - ❖ Reflect on what went well and what could be improved.
   - ❖ Identify any issues or challenges faced during the sprint.
   - ❖ Make plans to address these issues and improve processes for future sprints.